

CERTIFICATE OF MAILING BY EXPRESS  
MAIL

"EXPRESS MAIL" Mailing Label No.  
EK917656435 US

Date of Deposit: 12-28, 2000

## IMAGE SENSOR ARRAY READOUT FOR SIMPLIFIED

### IMAGE COMPRESSION

#### FIELD OF THE INVENTION

5           The invention relates generally to image processing and, more particularly, to the control of image sensor arrays.

#### BACKGROUND OF THE INVENTION

10           The success of 2D sensor development has permitted relatively high levels of integration, so that several blocks, such as timing control, PGA and ADC can be provided together with a 2D sensor array (for example a CMOS array) on a single integrated circuit chip. A conventional example of such a single chip imager 10 is illustrated in the conventional system diagram of FIGURE 1.

As the use of digital cameras, video conferencing and digital video has increased, a corresponding increase in the need for image compression has arisen. FIGURE 1 illustrates conventional components of an imaging compression architecture, including an image processing chip 13 which includes a compression engine 14, and a suitable memory chip 15. A multi-chip architecture such as shown in FIGURE 1 has several disadvantages, including the expense of the individual chips, the space requirements of the individual chips, and the total power consumption of the multi-chip architecture. These factors can vary with the size and complexity of the individual chips. For example, as the size and/or pixel resolution of the sensor array 12 within the imager 10 increases, the size of the memory chip 15 also increases.

Many conventional image compression algorithms process pixel elements in 8 x 8 blocks. Using a Discrete Cosine Transform (DCT), wherein each 8 x 8 block includes 8 row pixel elements and 8 column pixel elements. In conventional image processing arrangements, the image is read out in rows of pixels starting from the top (or bottom) of the sensor array and working down (or up). This requires the compression engine to store 8 complete rows of pixel data before it can actually begin compression operations. Moreover, because the image compression operations would, in general, be performed

concurrently with the process of reading pixels from the sensor, the compression engine would normally read out the next 8 rows of pixel data while compressing blocks from the previous 8 rows. Thus, for real time compression in this example, the memory of FIGURE 1 must be adequately sized to store 16 complete rows of pixels. Many  
5 conventionally available sensor arrays can include more than 3 million pixels, with over 2,000 pixels per row. Assuming a resolution of 8 bits/pixel, more than  $16 \times 2000 = 32,000$  bytes of memory would be needed to permit the compression engine to operate in real time.

FIGURE 2 illustrates a conventional example of a 2D CMOS sensor array. In the  
10 pixel detail of FIGURE 2, R\_SW designates a row reset switch, and S\_SW designates a row select switch. The row reset switches of the pixels of a single pixel row are connected together by metal lines, as are the row select switches of the row, such that both the row reset switches and the row select switches can be controlled by suitable control signals produced for that row by a row decoder. Examples of these control  
15 signals are shown in FIGURES 3 and 4. As shown therein, the pixels of each row receive a common row reset signal Row\_r(i) and a common row select signal Row\_s(i). The row reset signal for a given row controls the row reset switches of the pixels of that row, and

the row select signal for a given row controls the row select switches of the pixels of that row. FIGURES 3 and 4 also illustrate conventional column readout lines, designated as Col\_s(m) and Col\_s(n). These column readout lines are driven by the pixel elements during column readout.

5 It is desirable in view of the foregoing to provide for image compression without the aforementioned cost, space and power consumption disadvantages associated with conventional image compression arrangements.

The present invention provides readout control signals that permit the pixels of an image sensor array to be read out in blocks that are compatible with the operation of a  
10 desired image compression algorithm. This reduces the amount of memory required by the image compression algorithm, thereby advantageously permitting a more highly integrated image compression system with correspondingly reduced cost, space and power consumption requirements.

## BRIEF DESCRIPTION OF THE DRAWINGS

FIGURE 1 illustrates an example of a conventional image compression system architecture.

FIGURES 2-4 illustrate details of the sensor array of FIGURE 1.

5        FIGURE 5 illustrates pertinent portions of exemplary embodiments of an image sensor array according to the invention.

FIGURE 5A illustrates in detail a portion of the column lines of FIGURE 5.

FIGURE 6 illustrates the embodiments of FIGURE 5 on a pixel block level.

FIGURE 7 diagrammatically illustrates exemplary embodiments of a block  
10    readout controller according to the invention for providing the control signals of FIGURES 5 and 6.

FIGURE 8 illustrates exemplary operations which can be performed by the embodiments of FIGURES 5-7.

FIGURE 9 illustrates an exemplary embodiment of an integrated image  
15    compression system architecture according to the invention.

FIGURE 10 illustrates pertinent portions of further exemplary embodiments of an image sensor array according to the invention.

FIG. 10

## DETAILED DESCRIPTION

In the following discussion, a monochrome sensor with a dynamic range of 8 bits per pixel is assumed for clarity of exposition. However, it will be evident to workers in the art that the exemplary image compression system architecture described below is readily applicable to monochrome or color sensors, with higher or lower dynamic ranges.

FIGURE 5 illustrates pertinent portions of exemplary embodiments of an image sensor array according to the invention. In the array of FIGURE 5, each row of pixels is segmented into a plurality of segments, for example segment 1 and segment 2 of FIGURE 5. The row select and row enable signals for the pixels of each segment are driven by control logic associated with that segment. In FIGURE 5, the control logic for each segment includes a pair of two input AND gates designated generally as A1 and A2. For each segment, the output of the associated gate A1 drives the row reset signal for the pixels of that segment, and the output of the gate A2 drives the row select signal for the pixels of that segment. One input of each of the gates A1 and A2 is driven by the associated row enable signal. For example, each of the gates A1 and A2 of segment 1 and segment 2 of row 1 have one input driven by the row 1 enable signal, namely

Row(1). The gates A1 and A2 associated with the segments of row 2 similarly have one input driven by the row 2 enable signal, namely Row(2), and so on.

The gates A1 of segment 1 of each row are also driven by a reset signal associated with segment 1, namely Reset(1). Similarly, the gates A2 of segment 1 of each row are driven by a read signal associated with segment 1, namely Read(1). Reset and read signals Reset(2) and Read(2) associated with segment 2 of each row are similarly applied to the respective gates A1 and A2 of segment 2 of each row, and so on.

It can therefore be seen that the gates A1 and A2 of each individual segment permit that segment to be individually reset and read out on the associated column lines, without affecting any other segment in the array. Therefore, by suitably controlling the read, reset and row enable signals of FIGURE 5, pixel blocks of a desired size can be directly accessed and read out from the array via the column lines.

FIGURE 6 illustrates exemplary pixel blocks which can be accessed using the control signaling scheme of FIGURE 5. The row enable signals of FIGURE 6 are designated as Row(x), where x can range from 1 through r, where r is the number of rows in the sensor array. The reset and read signals of FIGURE 6 are designated Reset(y) and Read(y), where y can range from 1 through s, where s is the number of segments in each row of the array. The size of the pixel blocks is determined by the segment size and the

number of rows in each pixel block. A pixel block of size  $m \times n$  would include a given  $n$ -pixel segment (e.g. segment 1) in each of  $m$  rows.

FIGURE 7 diagrammatically illustrates exemplary embodiments of a block readout controller which provides the row enable, reset and read signals of FIGURES 5 and 6, and which also provides a correlated double sampling (CDS) signal, and suitable column multiplexing control 70 to permit the desired block readout via column multiplexer 72. The operation of the block readout controller 71 of FIGURE 7 is best understood with reference to the flow diagram of FIGURE 8.

The example of FIGURE 8 assumes  $s$  segments of  $n$  pixels in each row of an array of  $r$  rows. At 80, a row index  $x$  and a segment index  $y$  are each initialized to a value of 1. At 81, the signals  $\text{Row}(x)$  and  $\text{Reset}(y)$  are activated in order to charge the photodiodes of segment  $y$  of row  $x$ . At 82, the signal  $\text{Reset}(y)$  is deactivated in order to begin the exposure time. At 83, the signals  $\text{Row}(x)$ ,  $\text{Read}(y)$  and CDS are activated so the pixels of the current segment can drive the associated column lines and charge the associated CDS capacitors (see 52 in FIGURE 5A). At 84, the signal CDS is deactivated, and the signal  $\text{Reset}(y)$  is activated, thereby performing the correlated double sampling (CDS) operation (see also 51 in FIGURE 5A).

At 85, the controller 71 outputs appropriate column multiplexer control (see 70 in FIGURE 7) to sense the  $n$  pixels of segment  $y$  of row  $x$ . Thereafter at 86, it is determined whether or not the row index  $x$  has reached a multiple of  $m$ . If not, then the row index  $x$  is incremented at 87, and operations return to 81. On the other hand, if the row index  $x$  is determined at 86 to be a multiple of  $m$ , then it is determined at 88 whether the segment index  $y$  is equal to  $s$ . If not, then the segment index  $y$  is incremented at 89, and the row index  $x$  is reset to a value of  $x - (m-1)$  at 90. Operations thereafter return to 81. On the other hand, if the segment index  $y$  is equal to  $s$  at 81, it is determined at 91 whether the row index  $x$  is equal to  $r$ . If so, operations are completed. If not, then the segment index  $y$  is reset to a value of 1 at 92, and the row index  $x$  is incremented at 93, after which operations return to 81.

Although the example of FIGURE 8 reads out  $m \times n$  blocks moving across the columns of the first  $m$  rows of the array and thereafter the next  $m$  rows of the array, and so on, the block readout controller 71 of FIGURE 7 can be designed to provide  $m \times n$  block readout in any desired order of  $m \times n$  blocks.

-Due to the block readout capability provided by the sensor array structure of FIGURES 5-7, and assuming a monochrome system with a resolution of 8 bits per pixel, an image compression system architecture according to the invention requires only  $2 \times m$

x n bytes of memory for an m x n block-based compression algorithm such as DCT. Thus, when processing 8 x 8 blocks, the invention requires only 128 bytes of memory, as compared to the 32,000 bytes of memory required in the aforementioned conventional example. This reduced memory size permits all of the components of an image  
5 compression system to be integrated onto a single integrated circuit chip, as illustrated generally in FIGURE 9.

FIGURE 10 illustrates pertinent portions of another exemplary embodiment of an image sensor array according to the invention. The row segment illustrated in FIGURE 10 has a segment reset signal driven by AND gate A3. An image sensor array having this  
10 structure can be controlled in generally the same fashion as illustrated in FIGURE 8, except there is no Read(y) signal to be activated at 83. The image sensor array embodiment of FIGURE 10 also permits the above-described reduction in the memory requirement for the compression engine, and therefore enables the single chip integration described above with respect to FIGURE 9.

15 Although exemplary embodiments of the invention are described above in detail, this does not limit the scope of the invention, which can be practiced in a variety of embodiments.